Blockchain-based Sybil Attack Resistance Protocol

Stylianos Agapiou

Melchior Thambipillai

Introduction

- Goal is to enhance Peerster with a protocol to prevent Sybil attacks
- Global idea : nodes need to solve cryptopuzzles to join the network
- The puzzle should depend on all the nodes
- Should expire at some point
- Peers should be authenticated to verify they solved a puzzle

Related work

- Paper 1 : nodes organized in hierarchy tree and need to solve cryptopuzzles from leaf to root
- Paper 2 : each active node contributes to the creation of a global puzzle
- Our solution : thanks to the blockchain, combines the 2 advantages

Outline of the project

- Blockchain-based joining : peers need to mine blocks to join. The blockchain registers which peers are joined.
- P2P authentication : use digital signatures to authenticate peers and verify their appartenance to the blockchain

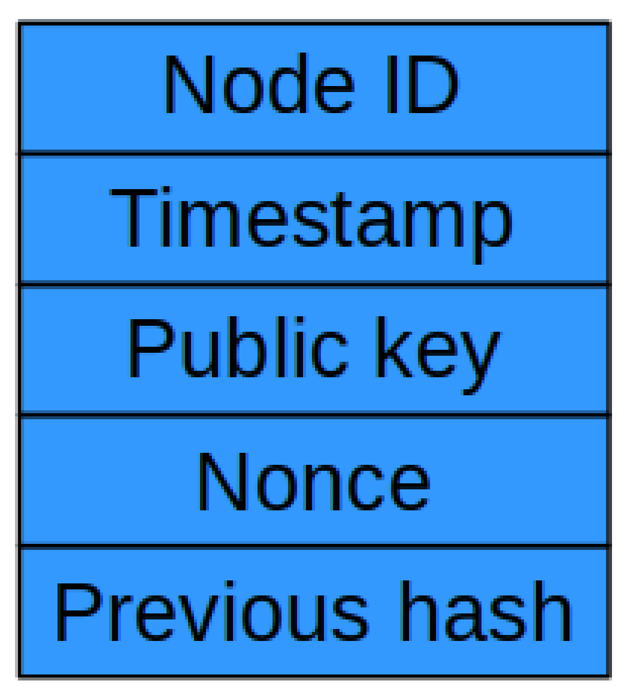
Blockchain-based joining

- Each block corresponds to one joined peer
- Steps of the protocol when A wants to join to B :

  1) B checks if A’s Node ID is valid in the blockchain

  2) if not, B sends a puzzle

  3) A mines a block and send it back to B



Node ID

Timestamp

Public key

Nonce

Previous hash

Blockchain-based joining

4) B verifies the block, add it to the blockchain and gossips a rumor with the new block

5) B sends the whole blockchain to A who is now joined

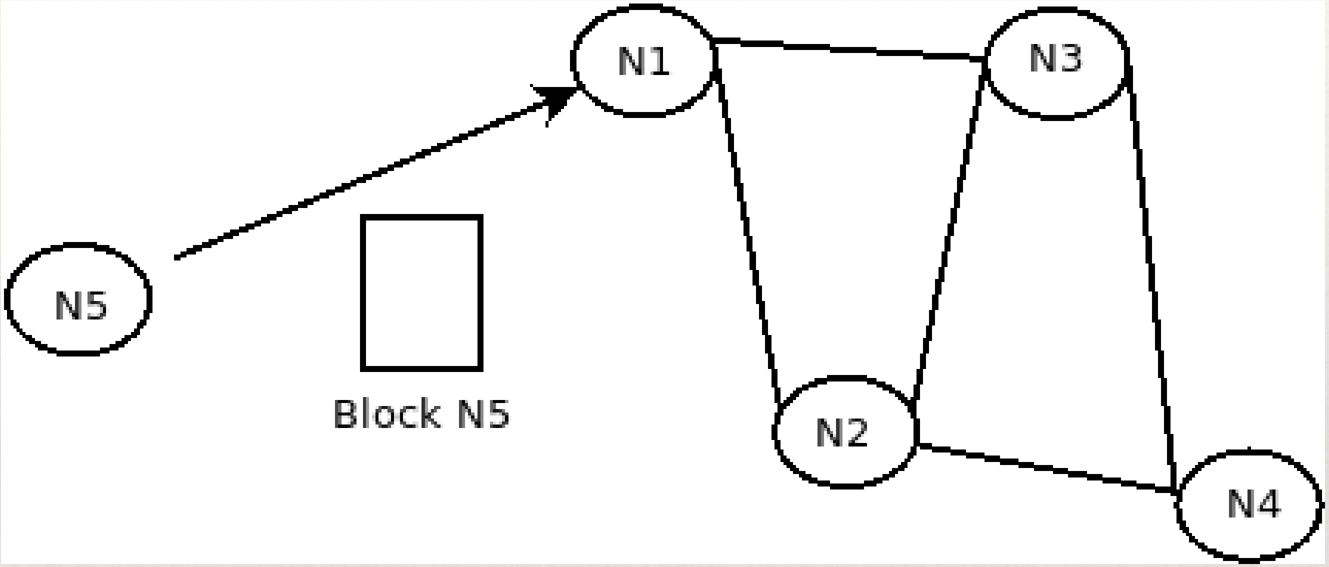
- Genesis : a user flag when creating a gossiper to start mining a genesis block or not
- Collisions : don’t need to handle them since we replace Node ID at every hop

P2P Authentication

- Digital signatures (RSA, SHA256)
- Inspection of inactive nodes

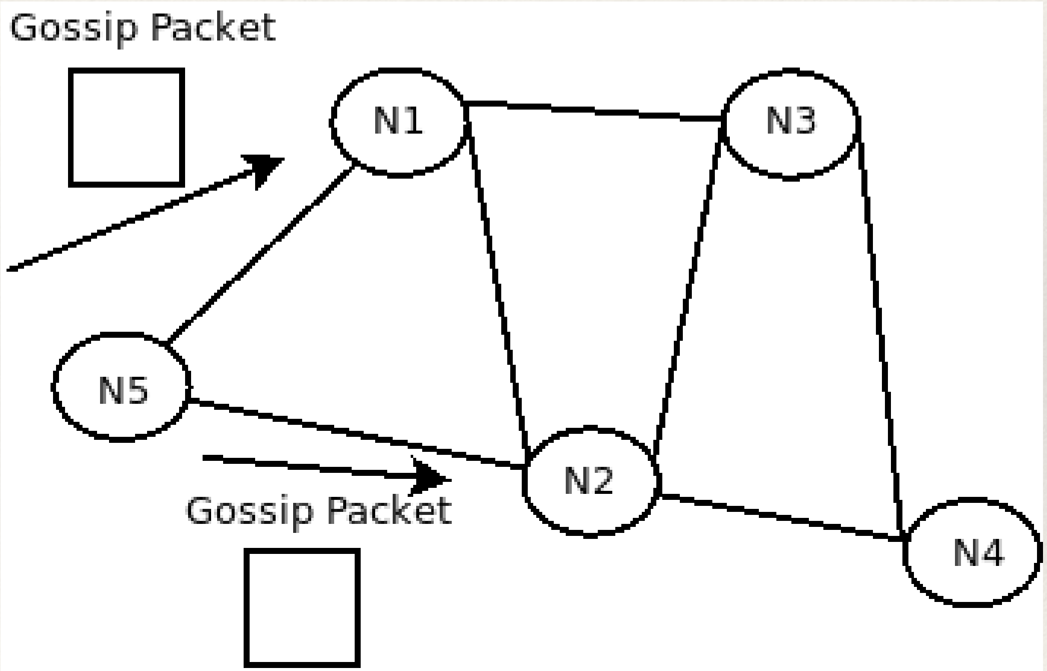
Digital Signature Algorithm

Each node creates an RSA key pair



Digital Signature Algorithm

- N5 hashes the message with SHA-256
- Then applies RSA to sign the message with it’s Private Key



Inspection of Inactive Nodes

- Use of Anti-Entropy mechanism
- If there’s no packet from a specific node for a significant amount of time
- We suspect this node as inactive and broadcast a signed packet to all neighbors

Evaluation

- Difficulty = 18

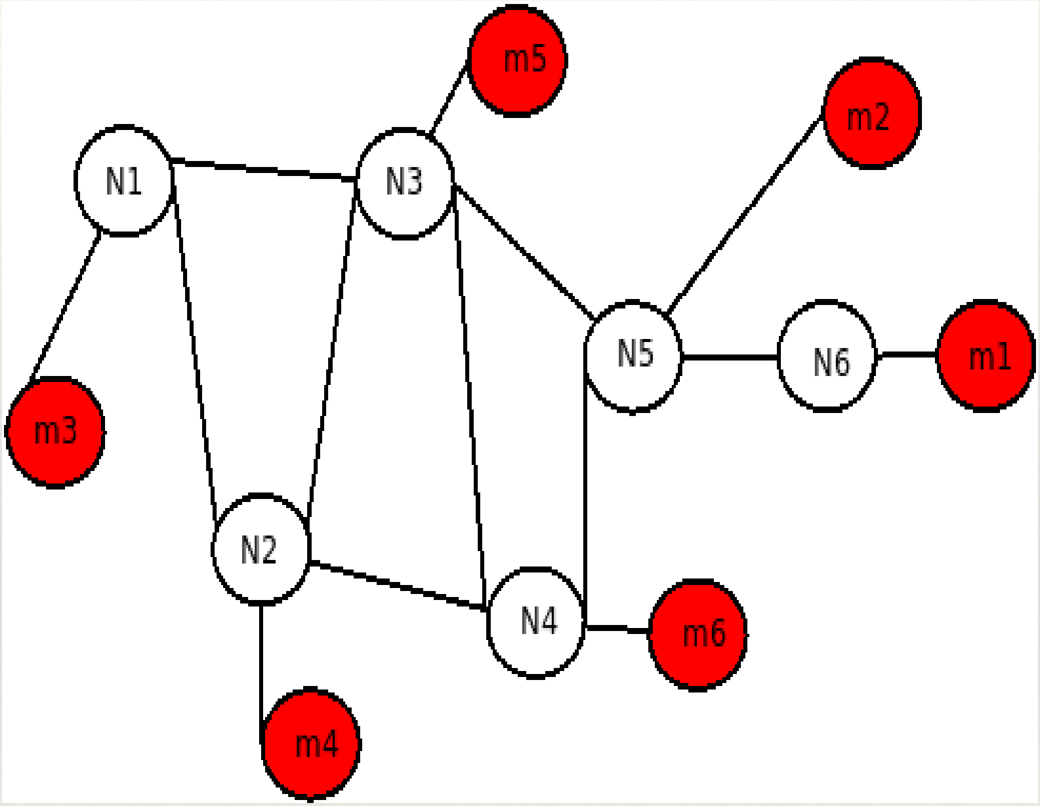
  1) 1.3468796s
  2) 768.457822ms
  3) 1.053998185s
  4) 1.581628689s
  5) 1.372276421s
  6) 1.472019263s

Difficulty = 20

1) 2.873795175s
2) 3.093252135s
3) 2.901926432s
4) 2.663215389s
5) 2.775930321s
6) 2.823405346s

Evaluation

With difficulty 24, it
took more than 10
minutes for nodes
(m1, m2,…, m7)
to join into
the blockchain



Conclusion

- Challenges
  1) Theory
  2) Build our project on top of Peerster

- Disadvantages
  1) PoW
  2) Handling of expiration

Conclusion

- Advantages of our solution
  1) Scalability
  2) Easy forks handling
  3) Completely decentralized

Thank you for your attention